IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application No.: 09/595,003	§	Examiner: Pillai, Namitha
Filed: June 13, 2000	§	Group/Art Unit: 2173
Inventor(s):	§	Atty. Dkt. No: 5150-44300
NICOLAS VAZQUEZ,	§	
JEFFREY L. KODOSKY,	§	
RAM KUDUKOLI,	§	
KEVIN L. SCHULTZ,	§	
DINESH NAIR, and	§	
CHRISTOPHE CALTAGIRONE	§	
Title: SYSTEM AND METHOD FOR AUTOMATICALLY GENERATING A GRAPHICAL PROGRAM TO IMPLEMENT A PROTOTYPE	§	

REPLY BRIEF

Sir or Madam:

Further to the Examiner's Answer of December 12, 2007, Appellant presents this Reply Brief, and respectfully requests that this Reply Brief be considered by the Board of Patent Appeals and Interferences.

STATUS OF CLAIMS

Claims 1-7, 9-37, 39-59, 61-74, and 76-90 are pending. Claims 8, 38, 60, and 75 are cancelled. Claims 1-7, 9-37, 39-59, 61-74, and 76-90 are rejected and are the subject of this Appeal.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Claims 1-7, 9-16, 21-37, 39-43, 45-59, 61-65, 67-74, and 76-90 are finally rejected under 35 U.S.C. 103(a) as being unpatentable over Morris et al. (U.S. Patent No. 5,862,372, hereinafter "Morris"), Oka et al. (EP Publication No. 0510514 A1, "Oka"), and Yamada (US Patent No. 4,831,580).

2. Claims 17-20, 44, and 66 are finally rejected under 35 U.S.C. 103(a) as being unpatentable over Morris, Oka, and Yamada, in further view of Shi et al. (U.S. Patent No. 5,623,659, hereinafter "Shi").

ARGUMENTS

The below is presented in response to the Examiner's Response to Argument in the Examiner's Answer. Appellant respectfully notes that the Examiner's Answer repeats arguments made earlier in the prosecution, which have been addressed in the Appeal Brief. Appellant has below presented responses to any *new* arguments in the Examiner's Answer. Thus, where no discernable new arguments were presented in the Examiner's Answer, Appellant has not added further responses, but relied on those presented in the Appeal Brief.

First Ground of Rejection:

Claims 1-7, 9-16, 21-37, 39-43, 45-59, 61-65, 67-74, and 76-90 are finally rejected under 35 U.S.C. 103(a) as being unpatentable over Morris, Oka, and Yamada. Appellant respectfully traverses this rejection, and presents the following additional reasons in response to the Examiner's new arguments. Different groups of claims are addressed under their respective subheadings.

Claims 1, 5, 7, 10, 13, 42, 53, 57, 59, 62, 64, and 90

Appellant respectfully submits that each of the independent claims 1, 53, and 90 recites one or more features not taught or suggested in Morris, Oka, and Yamada, taken singly or in combination.

The Examiner argues that "it is the combination of Morris, Oka, and Yamada that discloses the features of the present invention's claims." However, as Appellant has discussed in detail, *none* of these references discloses automatically generating a graphical program, and so Appellant respectfully submits that the references in combination cannot teach this feature.

For example, the Examiner has admitted that Morris does not disclose automatic generation of a graphical program (Morris discloses *manual* creation of a graphical program), but asserts that Oka and Yamada remedy this admitted deficiency of Morris.

In response to Appellant's argument that Oka discloses automatically generating a flowchart, not a graphical program, the Examiner asserts that "the disclosure of Oka provides motivation and suggests to a user the automatic generation of a program including automatic generation of nodes and connections of these nodes".

Appellant respectfully notes that the symbols/blocks in the flowchart generated by Oka are not graphical program nodes, are not executable, and are never described as such in Oka. In fact, Oka never mentions "nodes" as tall. Rather, Oka clearly describes the flowchart generated as an *illustration* of a pre-existing text-based program, e.g., that is printed out in lieu of a source code listing. Appellant submits that the Examiner has improperly characterized Oka's flowchart symbols and shapes as interconnected nodes, e.g., of an executable graphical program, which is technically incorrect. Applicant submits that generation of a picture and generation of an executable graphical program are very different things.

Appellant disagrees with the Examiner's assertion that Oka provides motivation and suggests to a user the automatic generation of a program including automatic generation of nodes and connections of these nodes, at least because Oka is not directed to, and nowhere teaches or suggests, or even mentions, graphical programs, graphical program nodes, or automatically generating programs of any type at all, nor does Oka indicate or hint at the desirability of automatically generating a graphical program, or any other type of program. Thus, a reader of Oka would not be motivated to consider automatic generation of graphical programs.

The Examiner asserts that "Morris upon learning from Oka would be motivated to automatically generate and connect the nodes to further alleviate user reliance for generating the graphical program which is a clear objective of Morris". Appellant disagrees, noting that Morris, being directed particularly to *manual* creation of a graphical program, and nowhere mentioning automatic generation of a graphical program, is not germane to such automatic generation. Nor would Oka's automatic generation of an illustrative flowchart of a pre-existing tex-based program motivate automation of Morris's manual process of creating a graphical program. As the Examiner is certainly aware, a flowchart generated based on an existing text-based program, and only illustrating the processing blocks of the program, in no way teaches or

suggests, or motivates automatically generating an executable graphical program (*not* from a previously existing text-based program). Thus, Oka also fails to disclose automatic generation of a graphical program.

The Examiner further asserts that Morris's intentions "are clearly to automate the program generation process". This is incorrect–as explained above and in the Appeal Brief, Morris is directed to manual creation of a graphical program, not automatic generation of a graphical program.

Regarding Yamada, the Examiner asserts that Yamada's flowcharts are executable, and are thus graphical programs. Appellant respectfully notes that Yamada nowhere describes the (manually created) flowchart as executable. Rather, in Yamada, once the user has created the flowchart, another program, a code generator, analyzes the flowchart and generates a text-based program based on the flowchart. Thus, the Examiner's assertion regarding Yamada's flowcharts is incorrect. Thus, Yamada also fails to disclose or hint at automatic generation of a graphical program.

The Examiner further asserts that Morris's manual selection of graphical program nodes and storing the nodes as a graphical program reads on Appellant's claimed "recording one or more functions in response to user input, wherein the one or more functions specify the algorithm". Appellant notes that equating these two processes is clearly at odds with the clear meaning of Appellant's claim 1, at least because if in claim 1 the user manually creates the graphical program (as described in Morris), there would be no point in automatically generating the graphical program. Thus, Morris does not teach or suggest this feature of claim 1.

Appellant thus respectfully submits that Morris's manual creation of a graphical program, combined with Oka's automatic generation of an illustrative flowchart from an existing text-based program, combined with Yamada's automatic generation of a text-based program from a manually created flowchart, in no way teaches or suggests, or motivates, automatic generation of a graphical program as claimed.

Claims 2 and 54

The Examiner argues that "the user dragging and dropping is a process which is interpreted as a function. The nodes each represent functions. In response to user input,

the process of dragging and dropping can be carried out. The dragging and dropping has to be recorded into the graphical program in order to generate the program and execute it."

This is incorrect on several counts. First, the Examiner appears to assert that "dragging and dropping" is a function, and also that each nodes represents a function, which implies that a node represents dragging and dropping, which is nonsensical. Appellant respectfully submits that it is quite clear from both the claims and the specification that "dragging and dropping" is not a "function" in the context of the present application.

Additionally, Appellant respectfully submits that in Morris, the "dragging and dropping" is not recorded, but rather results in the graphical program node being included in a graphical program.

Claims 11, 40, and 63

The Examiner fails to acknowledge that the palette is part of the development environment, *not* part of the graphical program, although this is clearly the case, as explained in the Appeal Brief. The Examiner also argues that "block diagram portions with connections between each other" are "nodes" that "represent user interface panel portions", and thus that "the user interface components represented as nodes make the graphical programs and reads on block diagrams and user interface panels".

Appellant respectfully submits that per Appellant's claims and specification, the block diagram and the user interface panel portions of Appellant's graphical program are separate and distinct, and that the graphical program nodes are comprised in the block diagram portion, *not* the user interface panel portion. Similarly, the user interface panel portion does not include graphical program nodes.

Appellant respectfully submits that one of skill in the art of graphical program development would readily understand that a palette from which one selects graphical program nodes for inclusion in a graphical program is not itself part of the graphical program.

Claim 12

The Examiner argues that Morris's graphical program represents a graphical data flow diagram where the graphical data includes nodes which flow through the diagram to carry out the functionality of the program." First, Appellant notes that the nodes do not flow through the diagram. More importantly, Appellant submits that "data flow" is a technical term in the field of computational models, and refers to the fact that nodes in a data flow diagram execute as soon as all their inputs are available. Thus, some graphical programs are not graphical data flow diagrams. Since Morris never mentions that the graphical program is a data flow diagram, Appellant submits that it is improper for the Examiner to read this attribute into Morris.

Claim 41

See the arguments for claim 12, above.

Claims 21, 45, and 67

The Examiner argues that the user selection of the node in Morris reads on user input specifying code generation information, and that based on the user's selections "code data is generated that is associated with the node that is selected". Again, the Examiner has failed to acknowledge that manual creation of a graphical program is not the same as automatic generation of a graphical program. Morris does not teach automatic generation of a graphical program, and thus cannot disclose automatically generating the graphical program utilizing code generation information specified by user input.

Claims 22, 23, 46, 47, and 68

"wherein the code generation information specifies a type of graphical program to create in response to the recorded one or more functions; wherein the graphical program is created in accordance with the specified graphical program type", as recited in claim 22.

The Examiner argues that if two graphical programs are different, e.g., in nodes, node placement, or node connections, they are of different graphical program types. This is incorrect. For example, two different cherry trees are of the same type, whereas a plum

tree and a cherry tree are of different types. In p.31:8-16 of the Specification, examples of different graphical program types are discussed:

> In step 322, input specifying a type of graphical program to create may be received. The input received in step 322 may specify a particular graphical programming language to use in generating the graphical program and/or may specify a particular graphical programming development environment where the generated program will be used. If a particular graphical programming development environment is specified in step 322, the generated graphical program may utilize proprietary programming or development features, or may adhere to proprietary requirements, of that specified graphical programming development environment.

Clearly, simply manually creating two different graphical programs in Morris's system does not mean that a graphical program type has been specified, and a graphical program automatically generated according to the specified type.

Claims 24, 48, and 69

The Examiner has mistakenly interpreted "during program operation" as "during *a* program operation". "During program operation" means "while the program is operating, i.e., executing", whereas "during *a* program operation" means during some operation relating to the program. Appellant respectfully submits that a program that is not executing is not "operating".

Claims 25, 49, 70

The Examiner continues to argue that Morris teaches automatic generation of a graphical program, when this is clearly not the case. Moreover, the Examiner argues that in Morris, "underlying code" is automatically generated in response to the user's creation of the graphical program. The Examiner goes on to argue that "the underlying code data that is automatically generated in Morris is linked together based on the structure of the graphical program", and that "Clearly the objective of Morris is automatic code generation based on the structure of a graphical flow diagram".

Appellant respectfully notes that as defined in Morris (col.4:19-22):

> "CODE" shall mean a series of specialized words and symbols in a particular syntax which must be entered through a keyboard and which is then converted to a machine readable form.

Additionally, col.5:26-31 reads:

> One of the interesting features of the object oriented authoring system of this invention is that no traditional looking code, written in a programming language, is generated by the system.

Appellant respectfully submits that linking portions of graphical program code together is quite different from linking automatically generated "underlying code data" based on the structure of a manually created graphical program. Additionally, as noted above, Morris does not disclose automatically generating "underlying code" in the first place. Nor does Morris even mention "link" or "linking" at all.

Claims 26, 27-29, 50 and 51

The Examiner fails to acknowledge that the "generating each portion of graphical code" is performed automatically, and thus, that the "connecting the node inputs and outputs together in order to implement the function with which the portion of graphical code is associated" is also performed automatically.

Since Morris only discloses *manual* creation of a graphical program, Morris does not, and cannot, disclose this feature.

Claims 30 and 52

The Examiner asserts that "mass storage describes a database which stores data required for the system to carry out the functionality", and that "the data stored in the database is used to generate a graphical program".

Morris never even mentions a database, and so cannot teach this feature.

Claim 4, 72, 82

The Examiner argues that "the output of the program includes image data where the graphical program including image processing to generate this output of an image in response to execution of the graphical program", but makes no citations.

Appellant notes that the only mention Morris makes of an "image" is in the Background, col.2:42-45, which reads:

> For instance, during the authoring process a view of the output generated by the application under development is useful when dealing with visual output (images).

Appellant further notes that simply generating an image as output does not mean "image processing", as is readily understood by those of skill in the art of image processing software.

Moreover, Morris nowhere even mentions the terms "image processing", "machine vision", "image analysis", "process control", "industrial automation", "test and measurement", "simulation", "workflow processes", or "robotics".

Claim 33

Morris nowhere even mentions prototyping, or a prototyping environment, and thus cannot teach this feature. Nor does Morris ever mention "image processing", "machine vision", "image analysis", "process control", "industrial automation", "test and measurement", "simulation", "workflow processes", or "robotics".

Claim 56

Morris nowhere even mentions prototyping, or a prototyping environment, and thus cannot teach this feature. Nor does Morris ever mention "image processing", "machine vision", "image analysis", "process control", "industrial automation", "test and measurement", "simulation", "workflow processes", or "robotics".

Claims 78, 87

The Examiner argues that "the graphical program with the nodes and the connections between the nodes represent a diagram model of the algorithm that is

generated." and that "this algorithm is implemented and executable as the graphical program in Morris".

Appellant respectfully reminds the Examiner that the prototype/diagrammatic model of the algorithm mentioned in claim 78 is distinct from the graphical program (see claim 71), and in fact, the graphical program is generated in response to the prototype. Thus, Morris's graphical program cannot be the prototype upon which the graphical program is based.

Second Ground of Rejection:

Claims 17-20, 44, and 66 are finally rejected under 35 U.S.C. 103(a) as being unpatentable over Morris, Oka, and Yamada, in further view of Shi et al. (U.S. Patent No. 5,623,659, hereinafter "Shi"). Different groups of claims are addressed under their respective subheadings. Appellant respectfully traverses this rejection, and presents the following additional reasons in response to the Examiner's new arguments. Different groups of claims are addressed under their respective subheadings.

Claims 17-20, 44, and 66

The Examiner asserts that "Shi does disclose preventing a user from editing the graphical program through a locking means". Appellant respectfully disagrees. As explained in the Appeal Brief, Shi is directed to locking/unlocking portions of a *data set* for editing by various users, and nowhere even mentions a "graphical program". Nor does Shi mention or even hint at locking an association between a script and a graphical program to prevent user editing of the graphical program. As the Examiner is certainly aware, a data set is not a graphical program, thus, Shi's data set locking/unlocking is not germane to the features of claim 17, and so Shi does not, and cannot, remedy the admitted deficiency of Morris, Oka, and Yamada.

CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-7, 9-37, 39-59, 61-74, and 76-90 was erroneous, and reversal of the decision is respectfully requested.

The Commissioner is authorized to charge any fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5150-44300/JCH.

Respectfully submitted,

/Jeffrey C. Hood/
Jeffrey C. Hood, Reg. #35,198
ATTORNEY FOR APPLICANT(S)

Meyertons Hood Kivlin Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800

Date: 2008-01-018 JCH/MSW